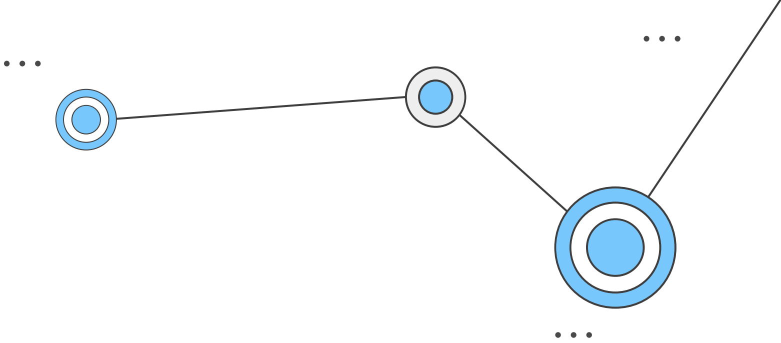
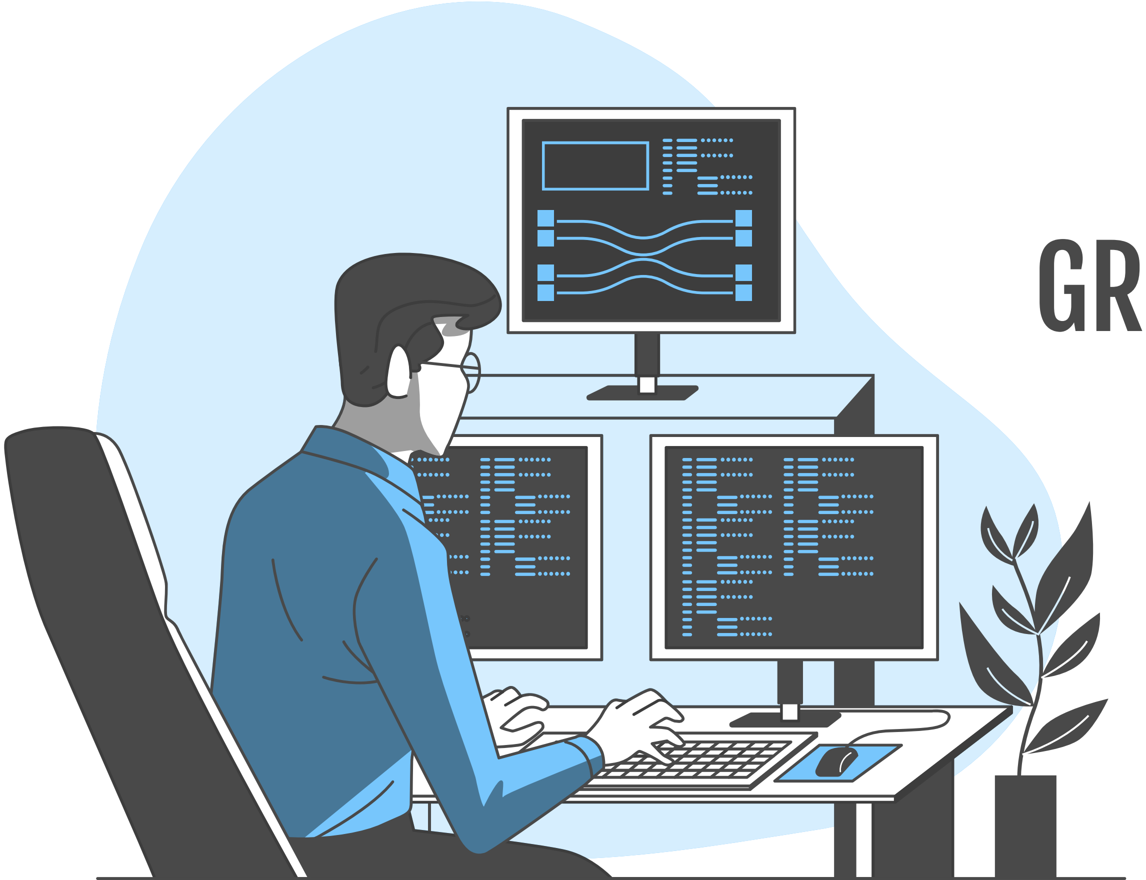




GRADLE AT SCALE IN ALLEGRO

With GitHub
and GitHub Actions!

WHY NOT MAVEN?

ALLEGRO USE CASES ARE MOSTLY SIMPLE

- declarative
- simple, easy to learn
- XML format is easy to analyse

OH, THAT'S IT!

I'VE JUST REMEMBERED THAT

- incremental builds?
- build caching?
- parallel task execution?
- IDE support?
- rich plugin system?
- flexibility?

GRADLE FTW!

👏👏👏

We decided to go with Gradle back in 2013
...and never looked back!

WE ARE ALLEGRO

SOME FACTS

- ~12k GitHub repositories (at least 2728 declare using Gradle)
- internal Gradle plugins created through the years
- internal maven repository
- multiple Gradle versions
- Gradle wrapper everywhere
- Gradle build cache server
- ~10k Gradle builds daily

THE CHALLENGE

EVERY COMPANY HAS THEM

We need to **know what dependencies** Allegro is built from.
We need to be **aware of security issues** related to supply chains.
We need to make sure that we are always **license compliant**.

CAUTIONARY TALES

THE CODE REPORT

Have you missed last week’s polyfill.io drama?

Remember when log4shell vulnerability striked?

THE PROBLEM

HOW TO OVERCOME IT?

With maven it is easy to collect all the dependencies that project uses.

Specialized tools like stackshare.io, fossa.com or GitHub
can simply parse maven xml file.

In gradle, the dependency graph is calculated during build...

PUZZLE MASTERS

FROM ANDAMIO TASKFORCE TEAM

Bartosz

Radek

WHAT ARE THE OPTIONS?

SOME RESEARCH NEEDED

Some companies develop internal gradle plugins
(like Nebula from Netlifx) so they can enforce some behaviours.

Some companies maintain their own gradle distributions.

Some try hard to parse and analyse gradle build files
(like fossa/stackshare does) - because it’s faster and does not rely on app build requirements itself .

Yet this way is prone to errors, since it’s static analysis of code.

GITHUB

POWERED BY DEPENDABOT

Github tries to build a Software Bill Of Material (SBOM) file out of any repository. In most languages it's easy and works well, but not with Gradle...

Github uses this data (at least) for:

- dependabot automated pull requests
- security advisories and reports

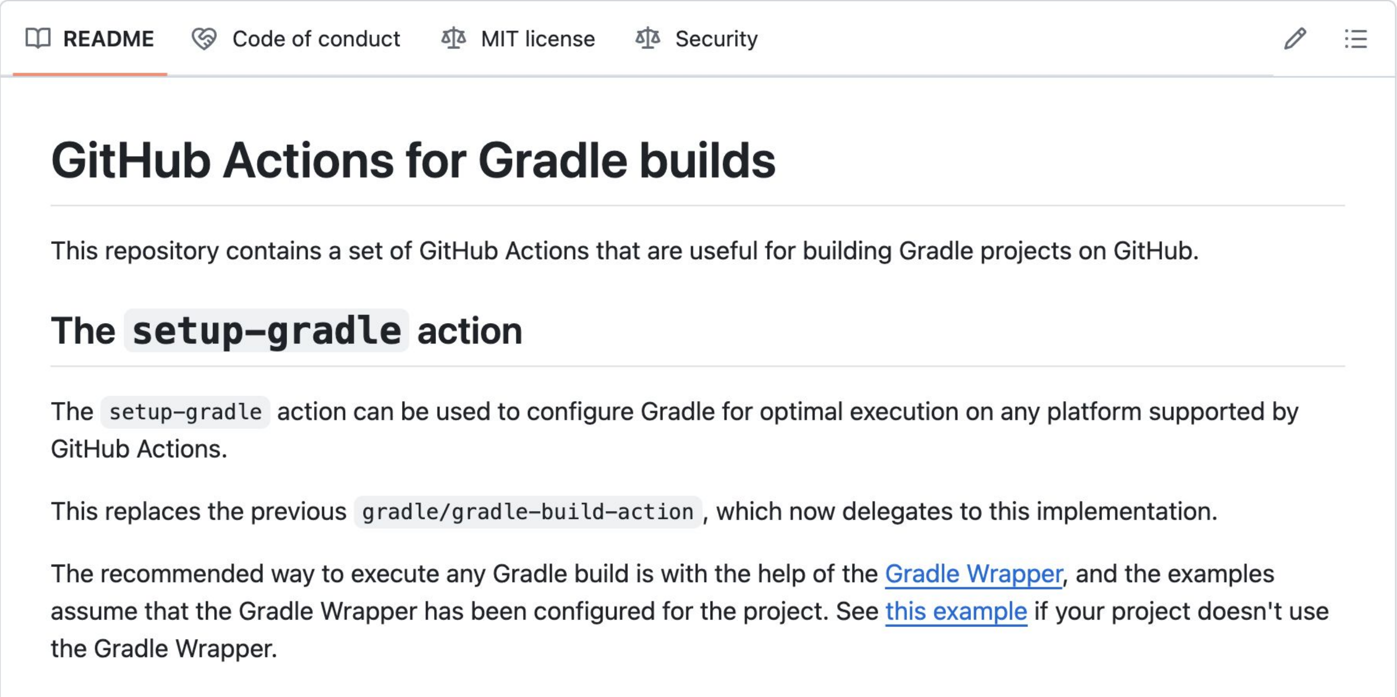
QUESTION

TO GRADLE, OR NOT TO GRADLE?

How to analyse Gradle application dependencies reliably than?

ACTIONS TO THE RESCUE!

GRADLE/ACTIONS/SETUP-GRADLE@v3



GitHub Actions for Gradle builds

This repository contains a set of GitHub Actions that are useful for building Gradle projects on GitHub.

The `setup-gradle` action

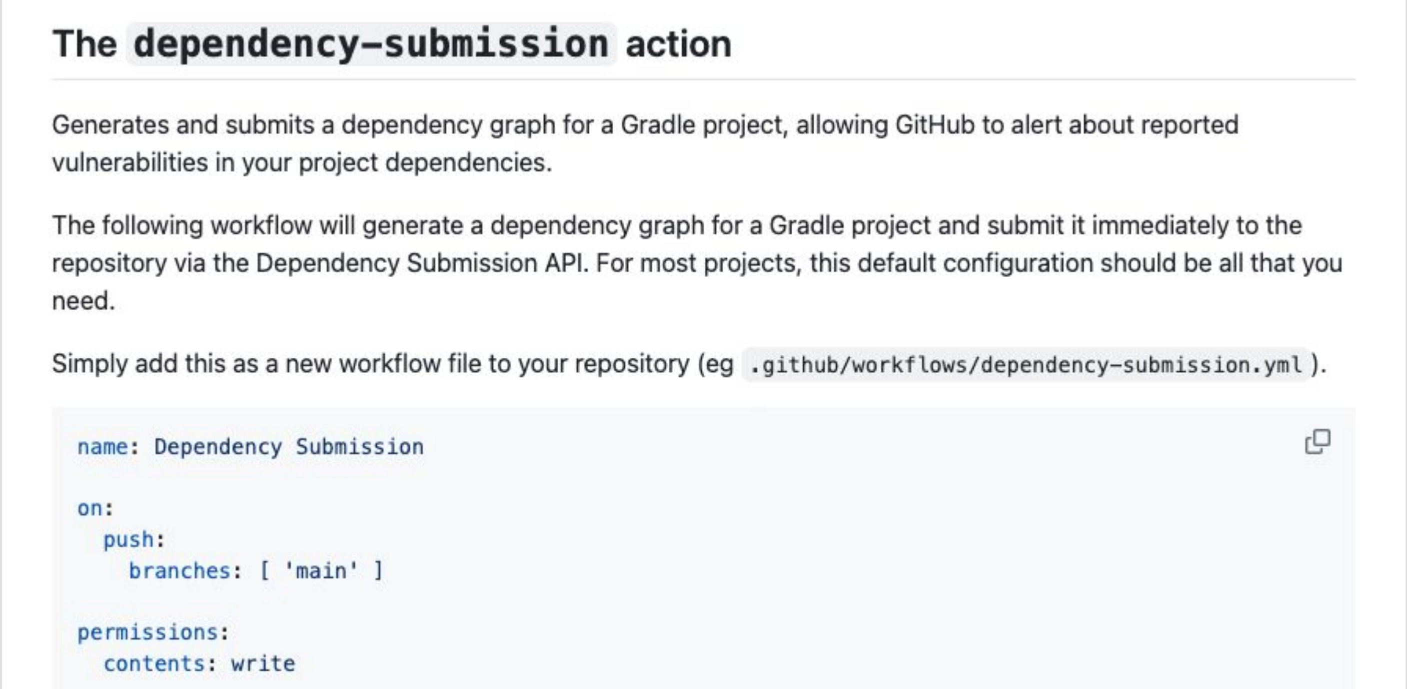
The `setup-gradle` action can be used to configure Gradle for optimal execution on any platform supported by GitHub Actions.

This replaces the previous `gradle/gradle-build-action`, which now delegates to this implementation.

The recommended way to execute any Gradle build is with the help of the Gradle Wrapper, and the examples assume that the Gradle Wrapper has been configured for the project. See this example if your project doesn't use the Gradle Wrapper.

ACTIONS TO THE RESCUE!

GRADLE/ACTIONS/SETUP-GRADLE@v3



The `dependency-submission` action

Generates and submits a dependency graph for a Gradle project, allowing GitHub to alert about reported vulnerabilities in your project dependencies.

The following workflow will generate a dependency graph for a Gradle project and submit it immediately to the repository via the Dependency Submission API. For most projects, this default configuration should be all that you need.

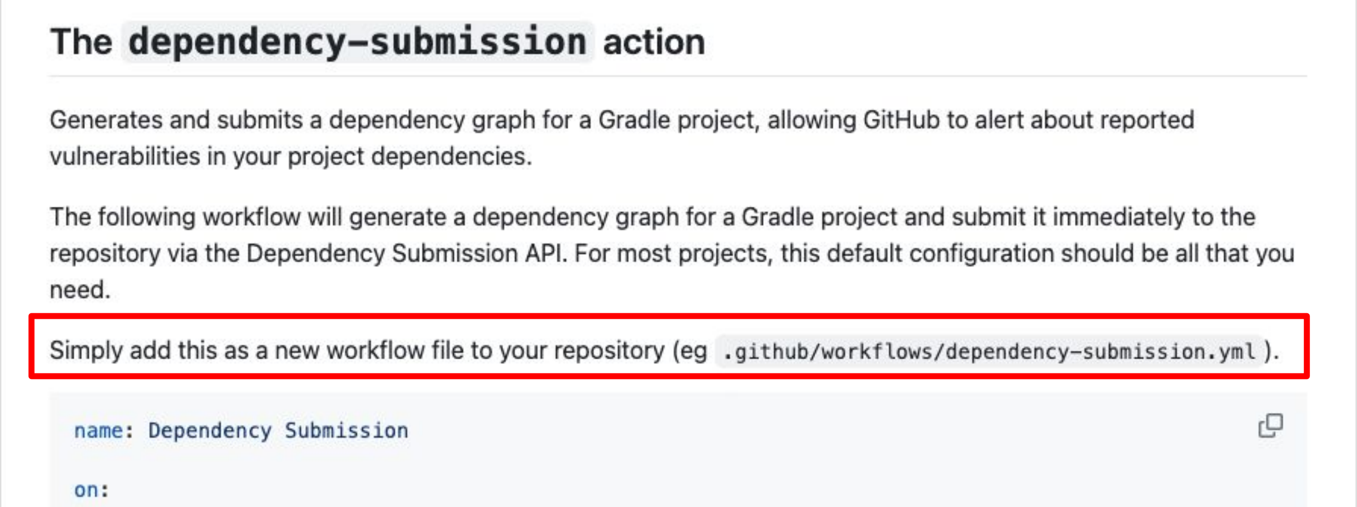
Simply add this as a new workflow file to your repository (eg `.github/workflows/dependency-submission.yml`).

```
name: Dependency Submission

on:
  push:
    branches: [ 'main' ]

permissions:
  contents: write
```

THAT GOT US THINKING...

WE ASKED OURSELVES SOME QUESTIONS!

Do we want to make a company-wide migration to add yet another workflow aside of existing ones? (some build's ain't that easy to set up)



The `dependency-submission` action

Generates and submits a dependency graph for a Gradle project, allowing GitHub to alert about reported vulnerabilities in your project dependencies.

The following workflow will generate a dependency graph for a Gradle project and submit it immediately to the repository via the Dependency Submission API. For most projects, this default configuration should be all that you need.

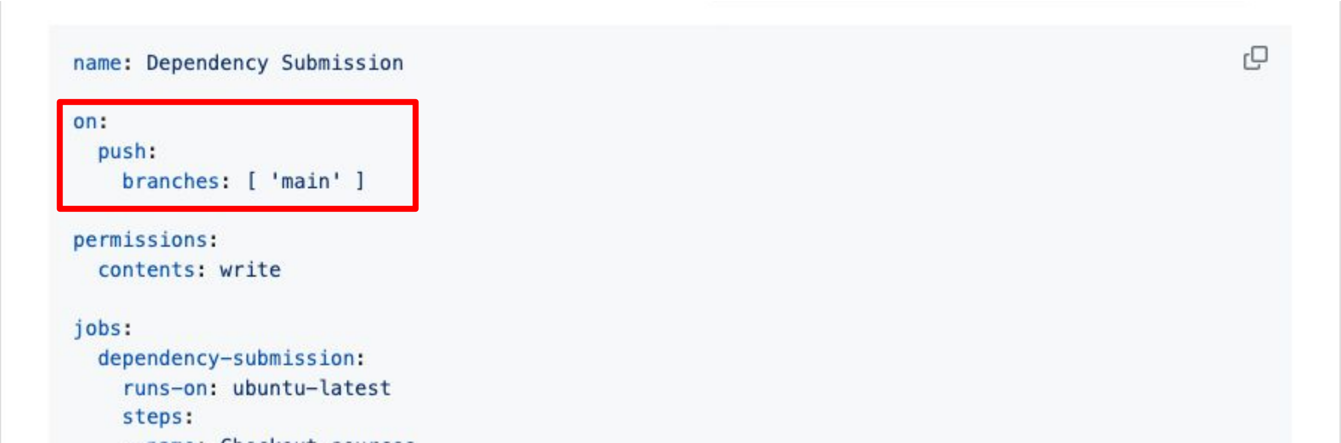
Simply add this as a new workflow file to your repository (eg `.github/workflows/dependency-submission.yml`).

```
name: Dependency Submission

on:
```

THAT GOT US THINKING...

WE ASKED OURSELVES SOME QUESTIONS!

There is no way to set push.branches: default branch trigger in GitHub.
We don’t want to hardcode “main” or “master” names...
shame on you GitHub!
(don’t worry, I still love you!)

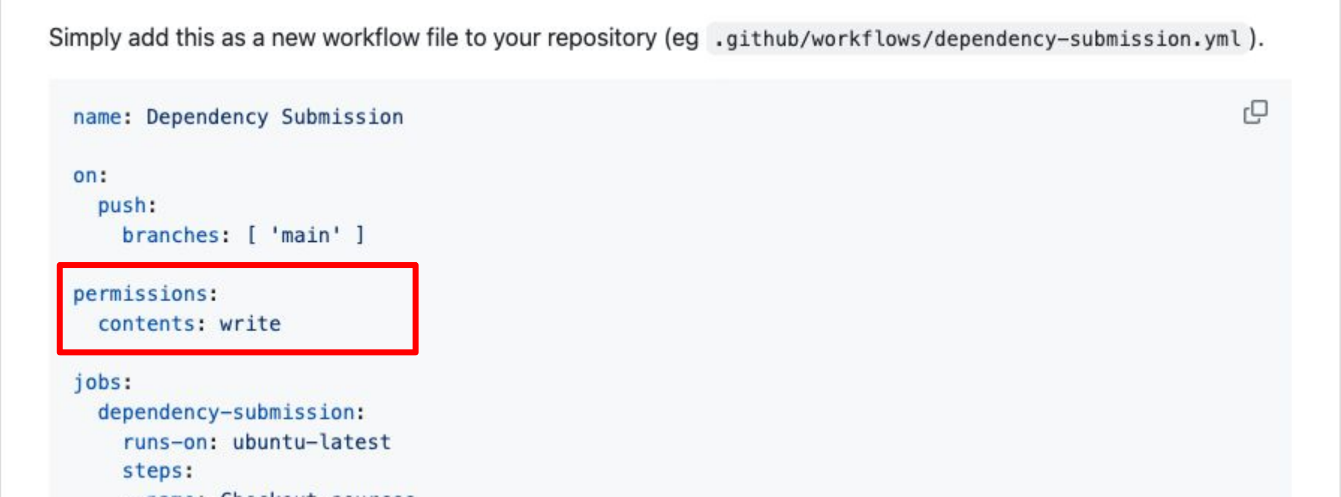


```yaml
name: Dependency Submission

on:
  push:
    branches: [ 'main' ]

permissions:
  contents: write

jobs:
  dependency-submission:
    runs-on: ubuntu-latest
    steps:
    - name: Checkout sources
```

THAT GOT US THINKING...

WE ASKED OURSELVES SOME QUESTIONS!

“Permissions: contents: write”
- how to deal with that if we don’t want additional workflow?



Simply add this as a new workflow file to your repository (eg `.github/workflows/dependency-submission.yml`).

```
name: Dependency Submission

on:
  push:
    branches: [ 'main' ]

permissions:
  contents: write

jobs:
  dependency-submission:
    runs-on: ubuntu-latest
    steps:
    - name: Checkout sources
```

OUR REQUIREMENTS

WHAT DO WE REALLY WANT?

We don’t want to enforce a reusable, universal build pipeline.

We want to have maximum customizability.

Developer experience and convenience is our priority.

WHY NOT ENABLE DEPENDENCY SUBMISSION GLOBALLY?

HOW COULD WE ACHIEVE THAT?

HOLD MY BEER

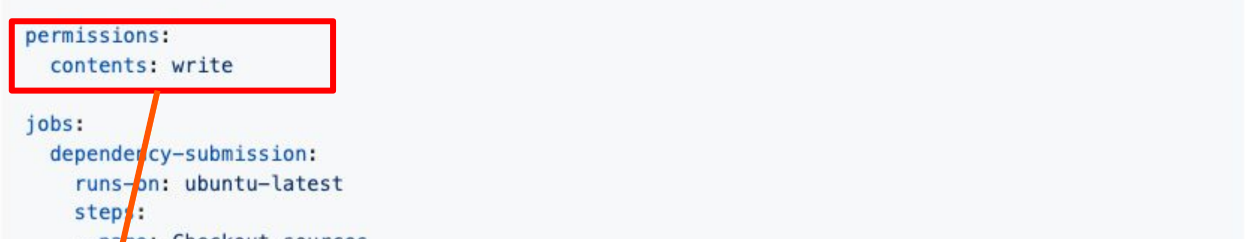
DOES IT MAKE SENSE?

Since we are going to do a gradle/actions/setup-gradle migration...

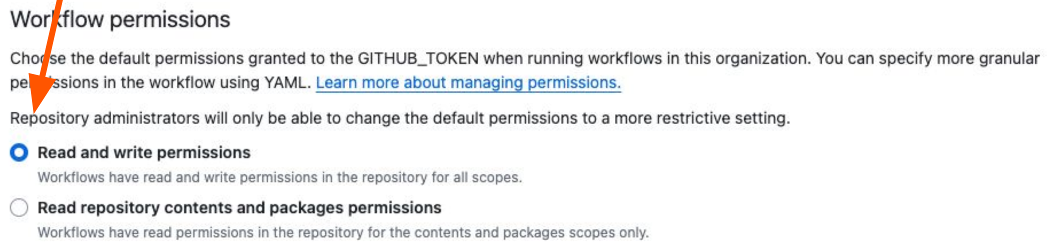
...we can wrap it in our own!

PERMISSIONS TURNABOUT

WE ALREADY HAVE ALL THE PERMISSIONS!



```
permissions:
  contents: write

jobs:
  dependency-submission:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout sources
```



Workflow permissions

Choose the default permissions granted to the GITHUB_TOKEN when running workflows in this organization. You can specify more granular permissions in the workflow using YAML. Learn more about managing permissions.

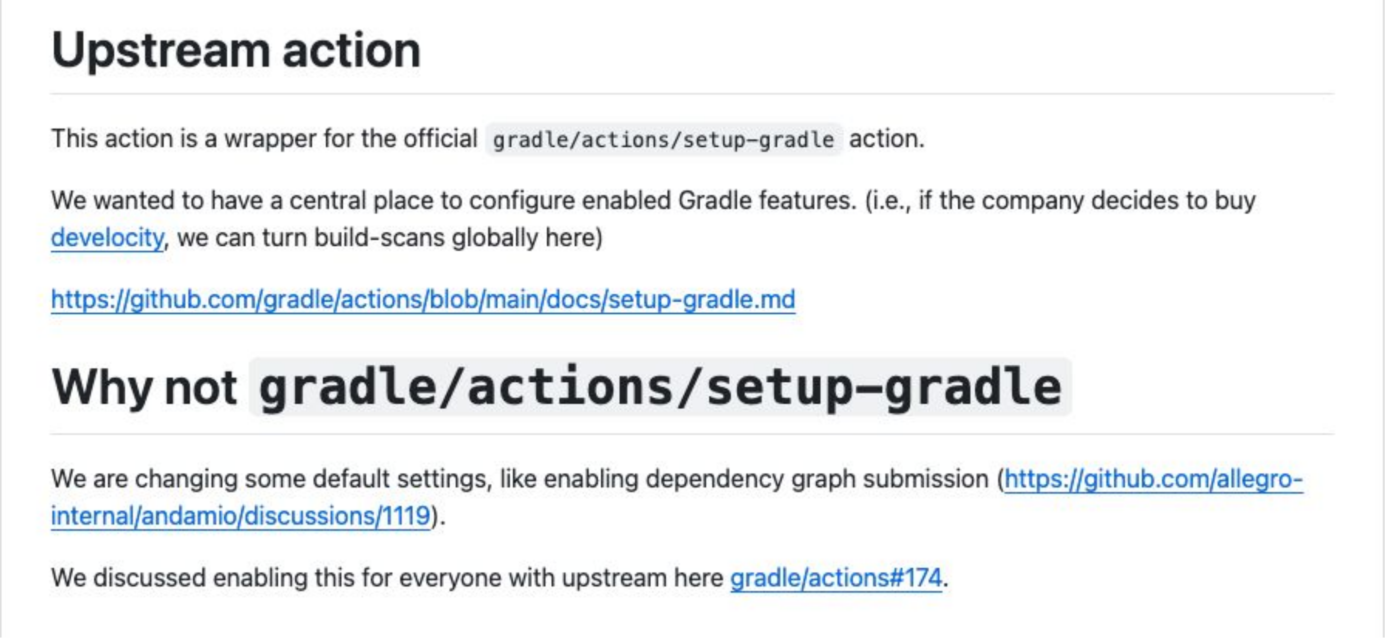
Repository administrators will only be able to change the default permissions to a more restrictive setting.

● **Read and write permissions**
Workflows have read and write permissions in the repository for all scopes.

○ **Read repository contents and packages permissions**
Workflows have read permissions in the repository for the contents and packages scopes only.

WE NEED TO GO DEEPER!

allegro-actions/setup-gradle@v1



Upstream action

This action is a wrapper for the official `gradle/actions/setup-gradle` action.

We wanted to have a central place to configure enabled Gradle features. (i.e., if the company decides to buy develocity, we can turn build-scans globally here)

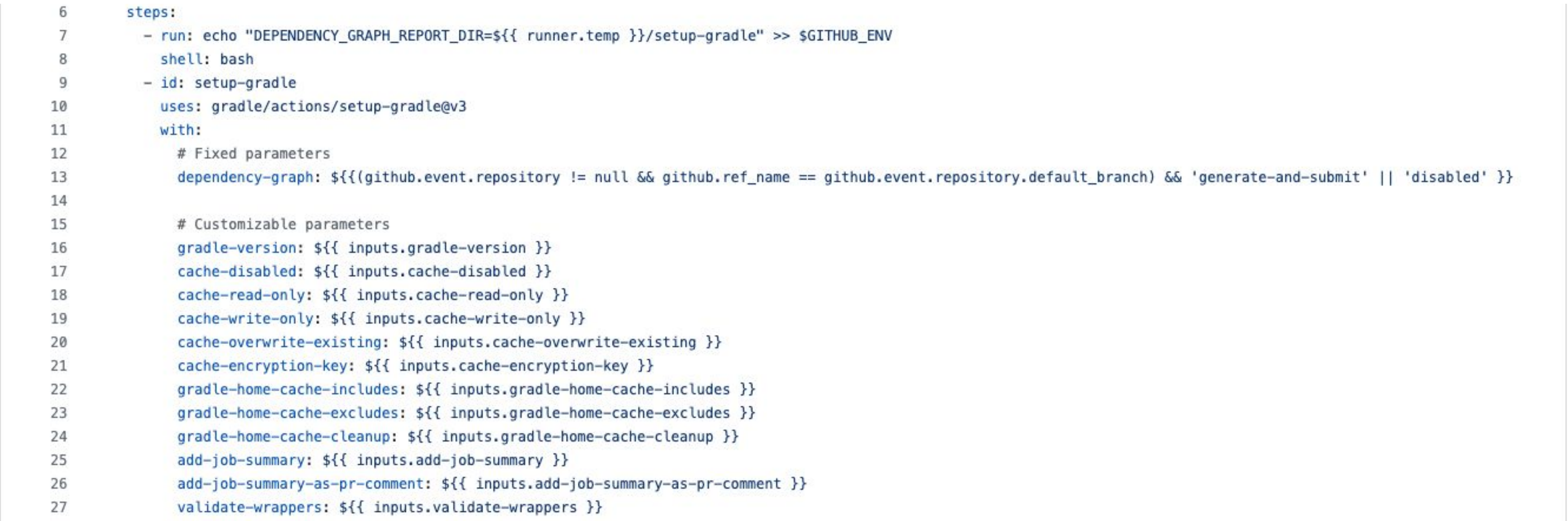
https://github.com/gradle/actions/blob/main/docs/setup-gradle.md

Why not `gradle/actions/setup-gradle`

We are changing some default settings, like enabling dependency graph submission (https://github.com/allegro-internal/andamio/discussions/1119).

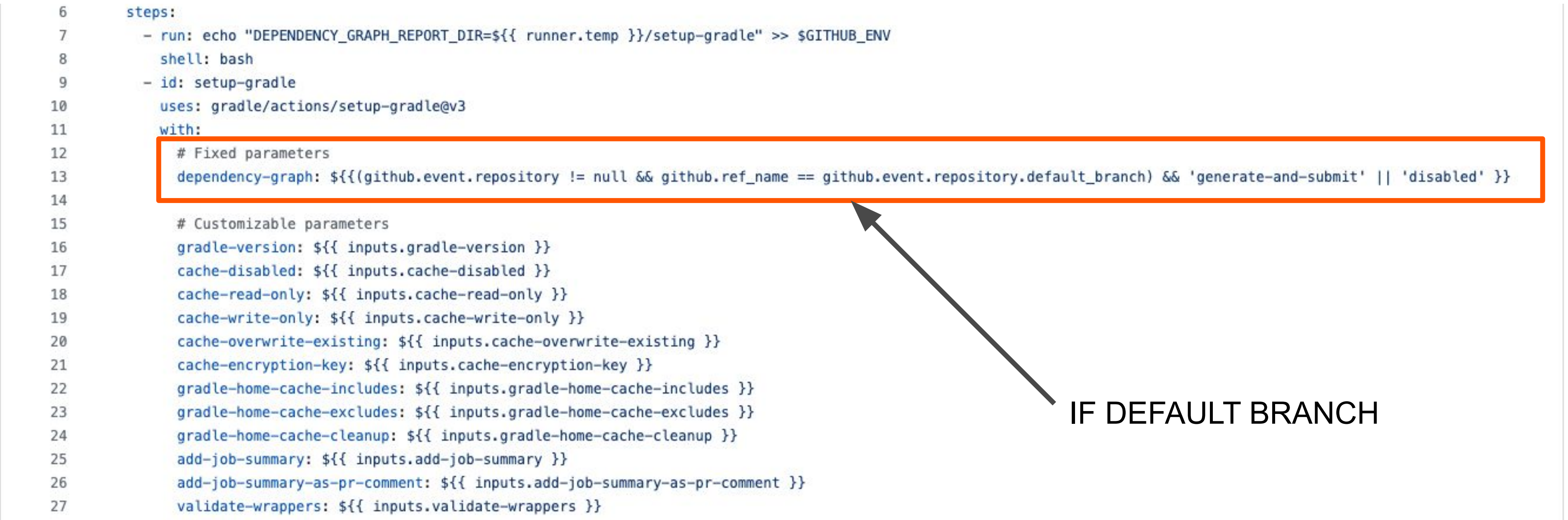
We discussed enabling this for everyone with upstream here gradle/actions#174.

27 MAGIC LINES
OF OUR GRADLE ACTION



```
    steps:
      - run: echo "DEPENDENCY_GRAPH_REPORT_DIR=${{ runner.temp }}/setup-gradle" >> $GITHUB_ENV
        shell: bash
      - id: setup-gradle
        uses: gradle/actions/setup-gradle@v3
        with:
          # Fixed parameters
          dependency-graph: ${{(github.event.repository != null && github.ref_name == github.event.repository.default_branch) && 'generate-and-submit' || 'disabled' }}

          # Customizable parameters
          gradle-version: ${{ inputs.gradle-version }}
          cache-disabled: ${{ inputs.cache-disabled }}
          cache-read-only: ${{ inputs.cache-read-only }}
          cache-write-only: ${{ inputs.cache-write-only }}
          cache-overwrite-existing: ${{ inputs.cache-overwrite-existing }}
          cache-encryption-key: ${{ inputs.cache-encryption-key }}
          gradle-home-cache-includes: ${{ inputs.gradle-home-cache-includes }}
          gradle-home-cache-excludes: ${{ inputs.gradle-home-cache-excludes }}
          gradle-home-cache-cleanup: ${{ inputs.gradle-home-cache-cleanup }}
          add-job-summary: ${{ inputs.add-job-summary }}
          add-job-summary-as-pr-comment: ${{ inputs.add-job-summary-as-pr-comment }}
          validate-wrappers: ${{ inputs.validate-wrappers }}
```

WE ENABLED DEPENDENCY SUBMISSION ON MAIN BRANCHES :)

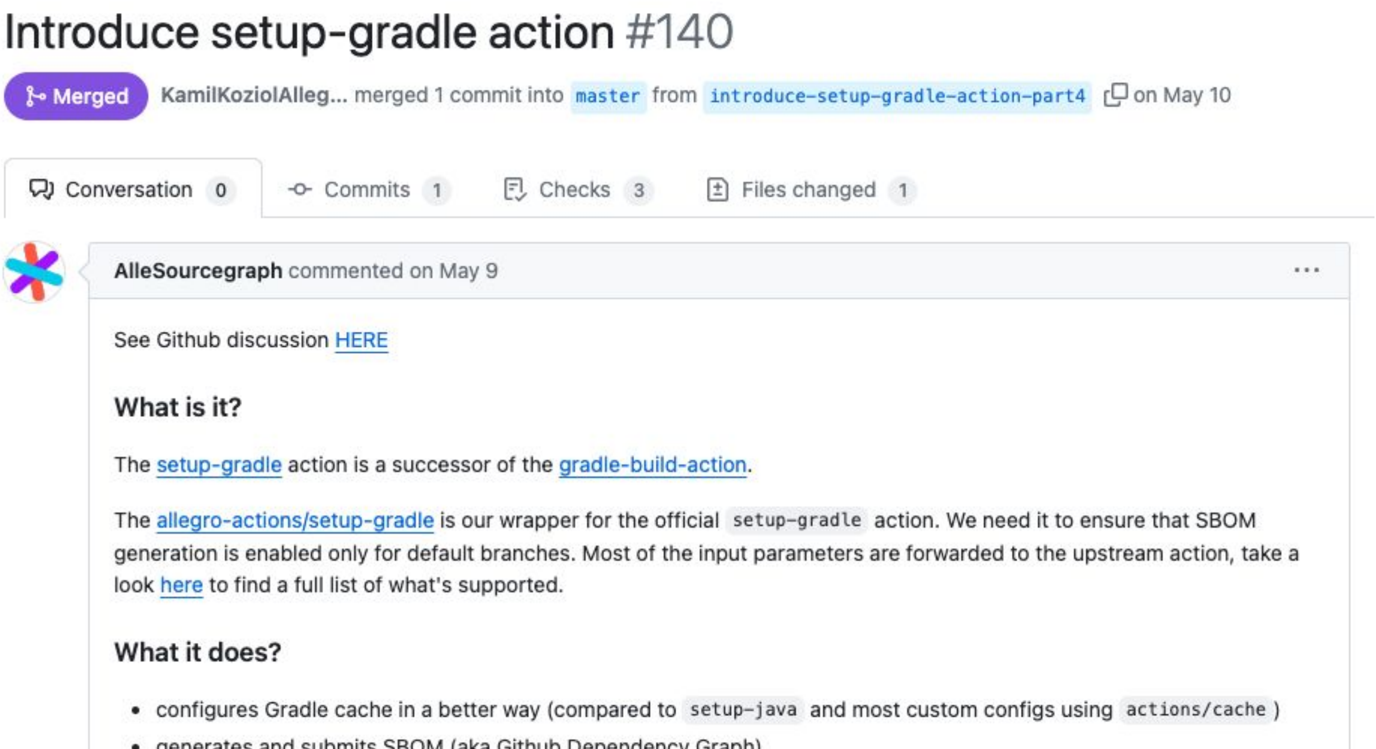


```
  steps:
    - run: echo "DEPENDENCY_GRAPH_REPORT_DIR=${{ runner.temp }}/setup-gradle" >> $GITHUB_ENV
      shell: bash
    - id: setup-gradle
      uses: gradle/actions/setup-gradle@v3
      with:
        # Fixed parameters
        dependency-graph: ${{(github.event.repository != null && github.ref_name == github.event.repository.default_branch) && 'generate-and-submit' || 'disabled' }}

        # Customizable parameters
        gradle-version: ${{ inputs.gradle-version }}
        cache-disabled: ${{ inputs.cache-disabled }}
        cache-read-only: ${{ inputs.cache-read-only }}
        cache-write-only: ${{ inputs.cache-write-only }}
        cache-overwrite-existing: ${{ inputs.cache-overwrite-existing }}
        cache-encryption-key: ${{ inputs.cache-encryption-key }}
        gradle-home-cache-includes: ${{ inputs.gradle-home-cache-includes }}
        gradle-home-cache-excludes: ${{ inputs.gradle-home-cache-excludes }}
        gradle-home-cache-cleanup: ${{ inputs.gradle-home-cache-cleanup }}
        add-job-summary: ${{ inputs.add-job-summary }}
        add-job-summary-as-pr-comment: ${{ inputs.add-job-summary-as-pr-comment }}
        validate-wrappers: ${{ inputs.validate-wrappers }}
```

IF DEFAULT BRANCH

NOW THE HARD PART

LET’S DO A BATCH CHANGE!



Introduce setup-gradle action #140

Merged KamilKoziolAlleg... merged 1 commit into `master` from `introduce-setup-gradle-action-part4` on May 10

AlleSourcegraph commented on May 9

See Github discussion HERE

What is it?

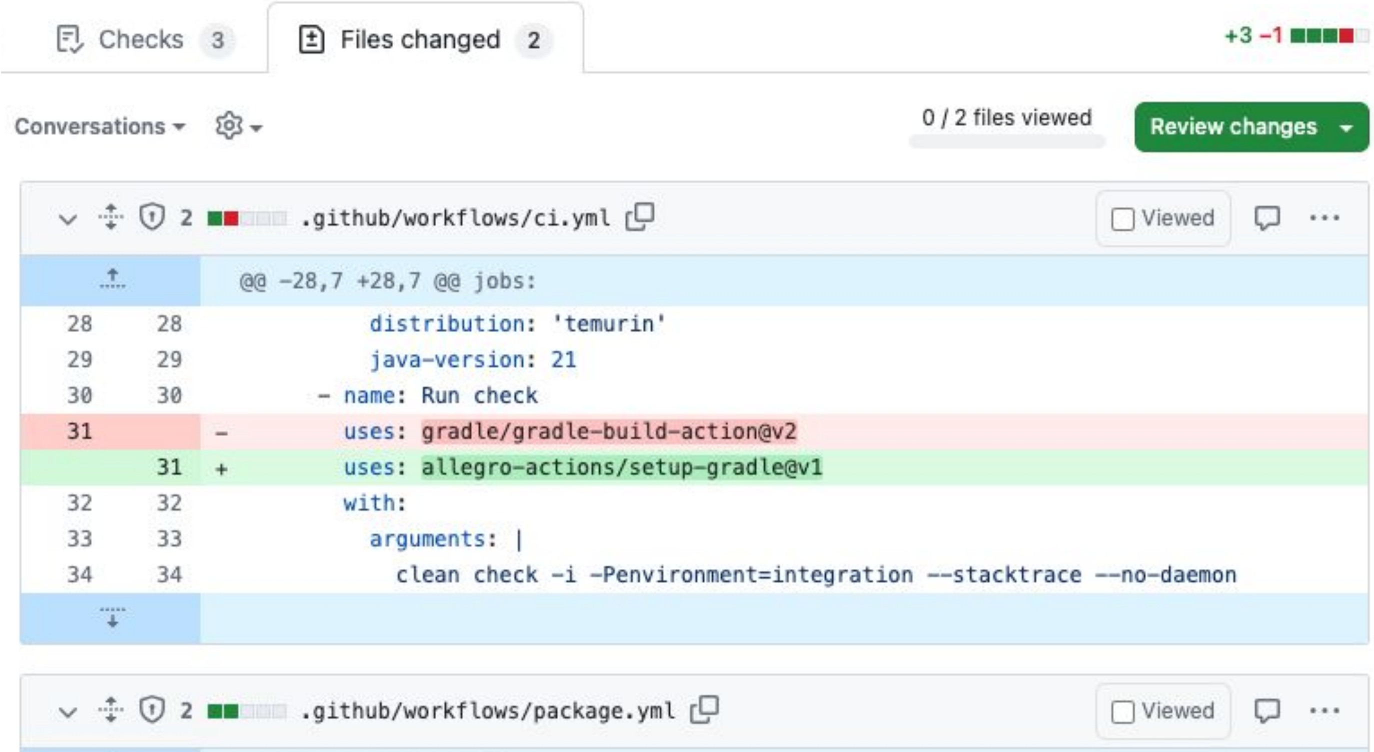
The setup-gradle action is a successor of the gradle-build-action.

The allegro-actions/setup-gradle is our wrapper for the official `setup-gradle` action. We need it to ensure that SBOM generation is enabled only for default branches. Most of the input parameters are forwarded to the upstream action, take a look here to find a full list of what's supported.

What it does?

- configures Gradle cache in a better way (compared to `setup-java` and most custom configs using `actions/cache`)
- generates and submits SBOM (aka Github Dependency Graph)

LET’S DO A BATCH CHANGE!

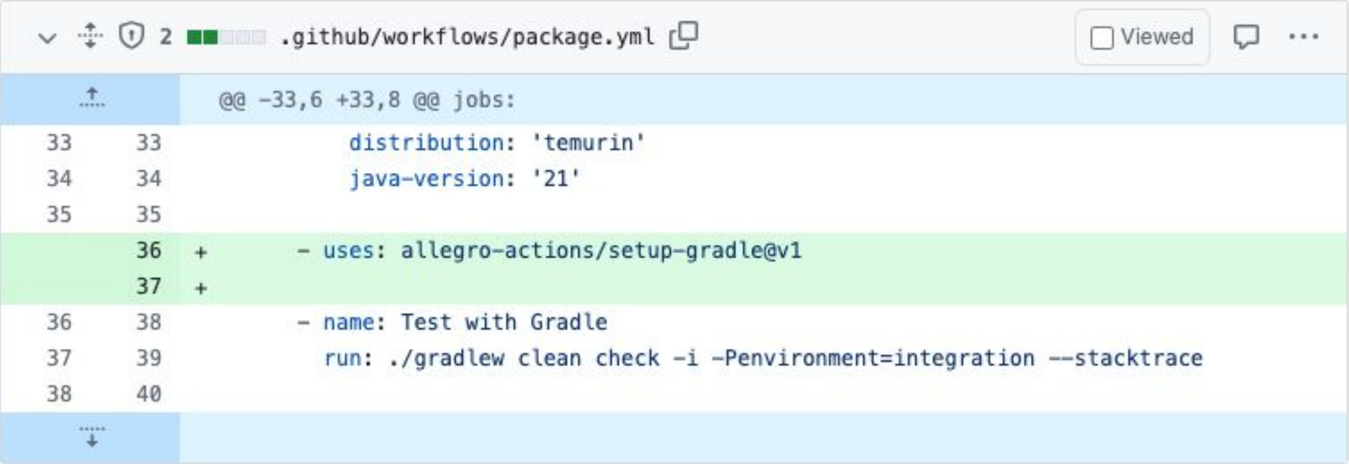


.github/workflows/ci.yml

```
@@ -28,7 +28,7 @@ jobs:
           distribution: 'temurin'
           java-version: 21
         - name: Run check
-          uses: gradle/gradle-build-action@v2
+          uses: allegro-actions/setup-gradle@v1
           with:
             arguments: |
               clean check -i -Penvironment=integration --stacktrace --no-daemon
```

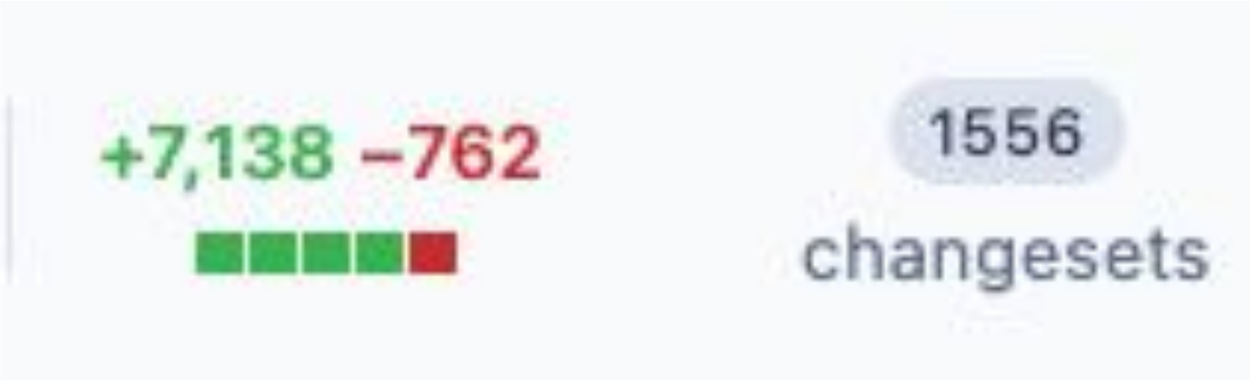
.github/workflows/package.yml

LET’S DO A BATCH CHANGE!



```
@@ -33,6 +33,8 @@ jobs:
          distribution: 'temurin'
          java-version: '21'

+      - uses: allegro-actions/setup-gradle@v1
+
       - name: Test with Gradle
         run: ./gradlew clean check -i -Penvironment=integration --stacktrace

```

SMALL CHANGE, I PROMISE



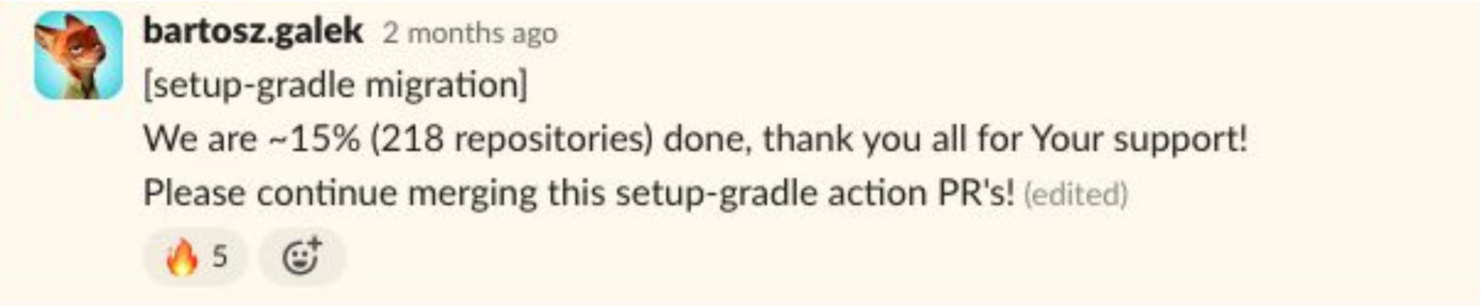
+7,138 −762

🟩🟩🟩🟩🟥

1556

changesets

IT WAS PAINFUL PROCESS...



bartosz.galek 2 months ago

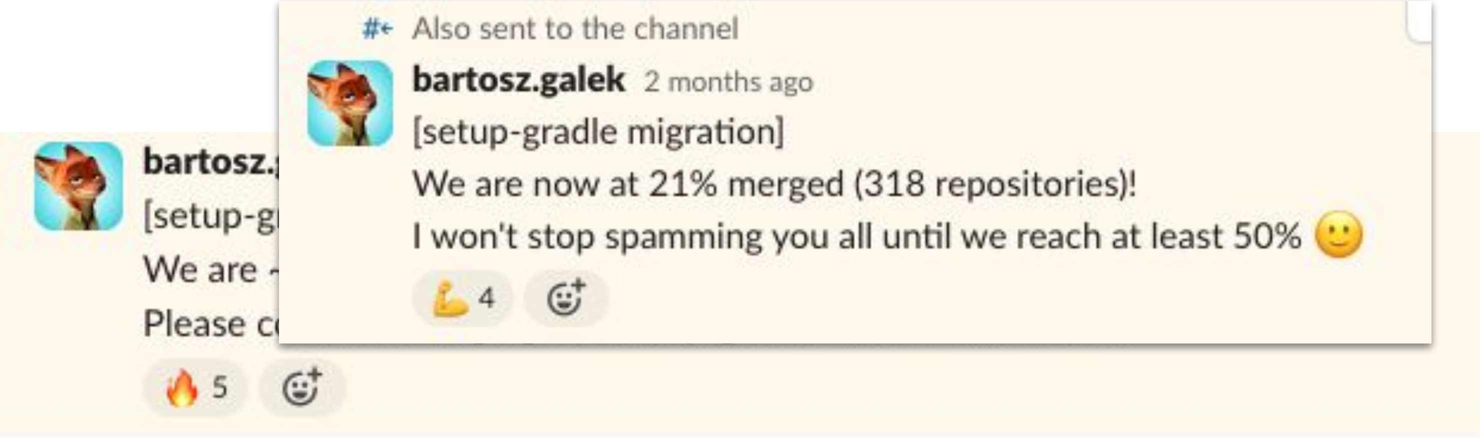
[setup-gradle migration]

We are ~15% (218 repositories) done, thank you all for Your support!

Please continue merging this setup-gradle action PR's! (edited)

🔥 5

IT WAS PAINFUL PROCESS...



bartosz.

[setup-g

We are -

Please c

🔥 5

#← Also sent to the channel

bartosz.galek 2 months ago

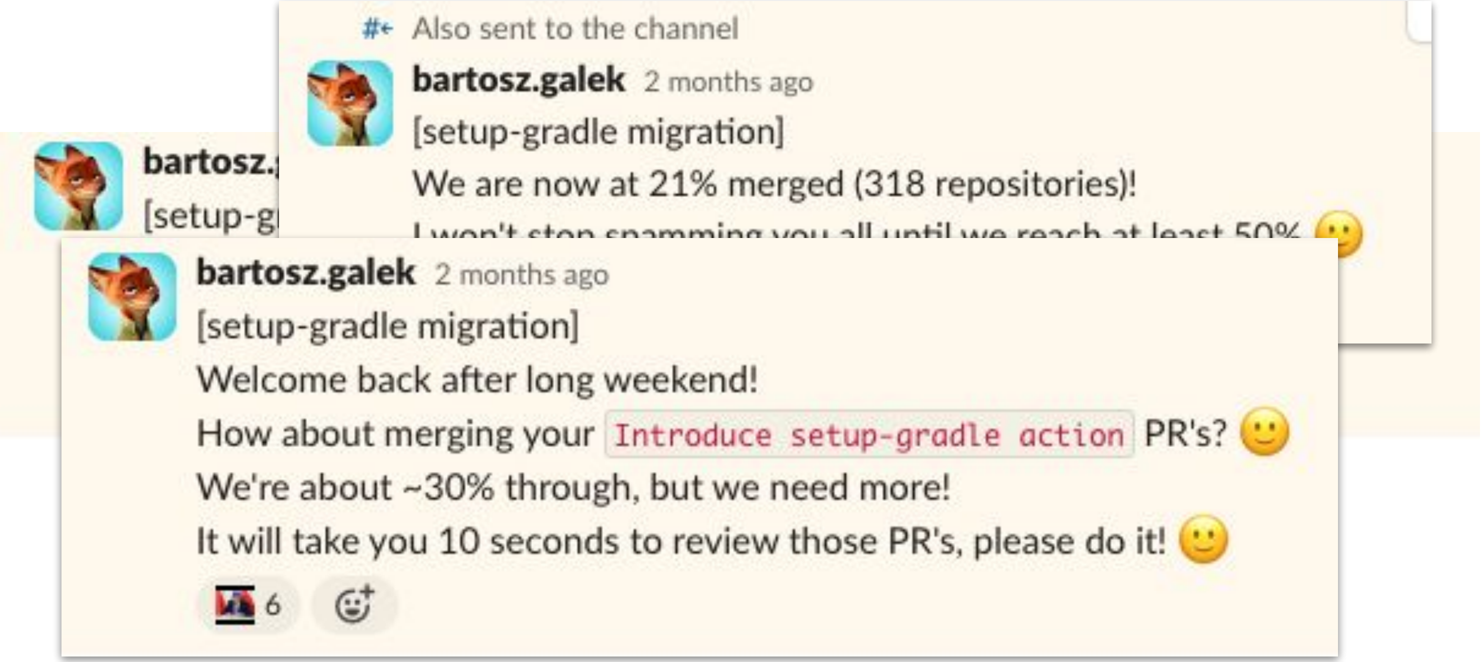
[setup-gradle migration]

We are now at 21% merged (318 repositories)!

I won't stop spamming you all until we reach at least 50% 🙂

💪 4

IT WAS PAINFUL PROCESS...



bartosz.

[setup-g

#← Also sent to the channel

bartosz.galek 2 months ago

[setup-gradle migration]

We are now at 21% merged (318 repositories)!

I won't stop spamming you all until we reach at least 50% 🙂

bartosz.galek 2 months ago

[setup-gradle migration]

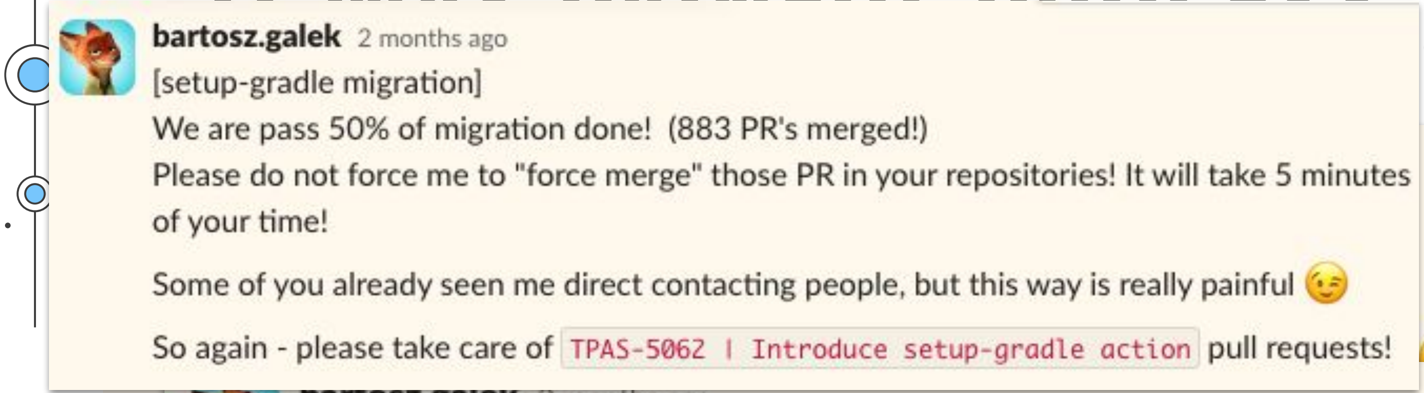
Welcome back after long weekend!

How about merging your `Introduce setup-gradle action` PR's? 🙂

We're about ~30% through, but we need more!

It will take you 10 seconds to review those PR's, please do it! 🙂

IT WAS PAINFUL PROCESS...



bartosz.galek 2 months ago

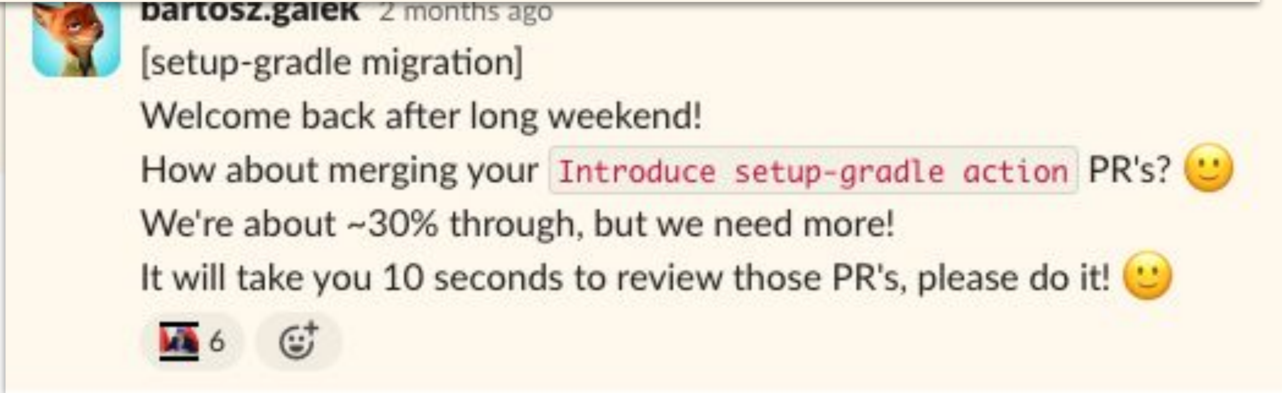
[setup-gradle migration]

We are pass 50% of migration done! (883 PR's merged!)

Please do not force me to "force merge" those PR in your repositories! It will take 5 minutes of your time!

Some of you already seen me direct contacting people, but this way is really painful 😉

So again - please take care of `TPAS-5062 | Introduce setup-gradle action` pull requests!



bartosz.galek 2 months ago

[setup-gradle migration]

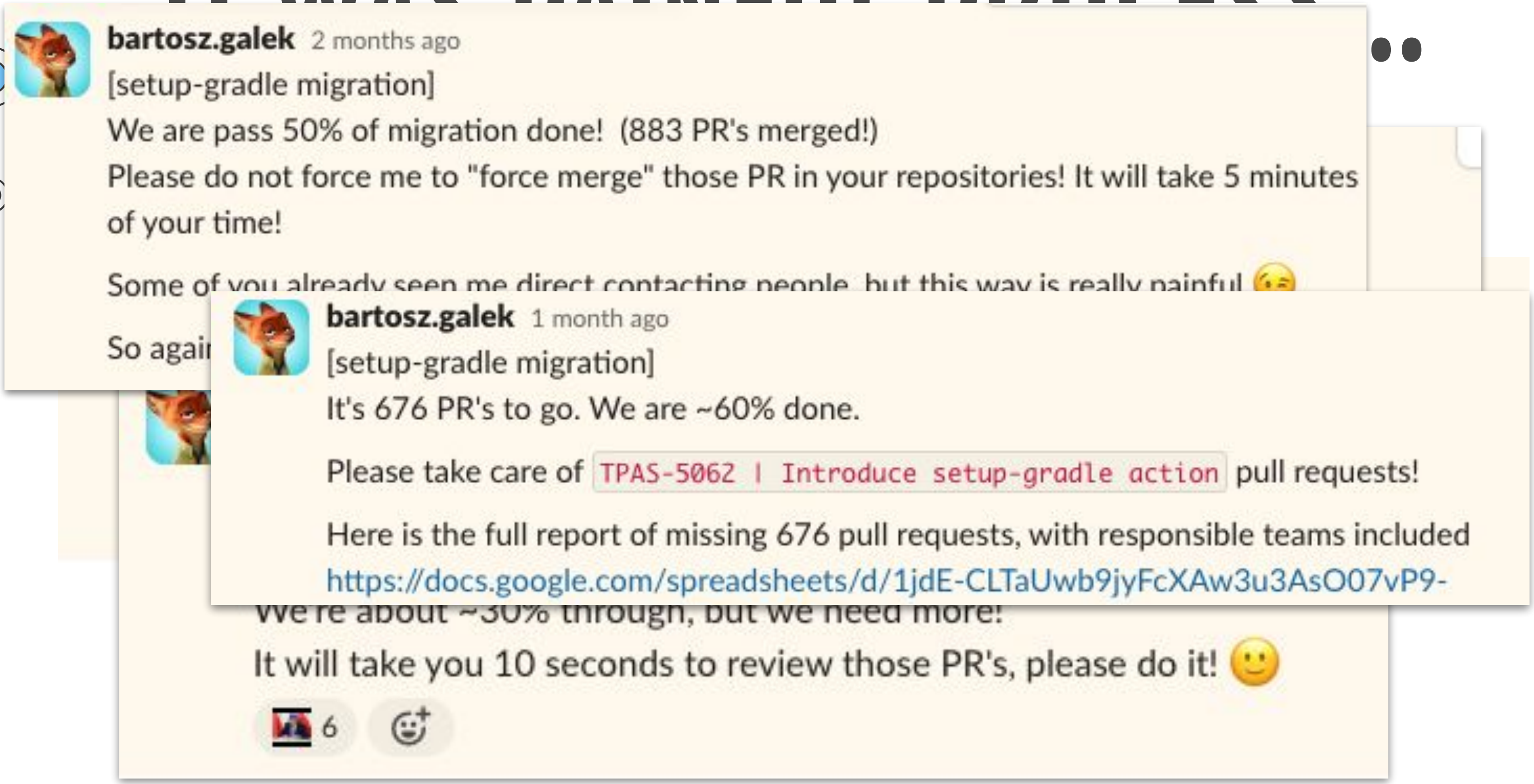
Welcome back after long weekend!

How about merging your `Introduce setup-gradle action` PR's? 🙂

We're about ~30% through, but we need more!

It will take you 10 seconds to review those PR's, please do it! 🙂

IT WAS PAINFUL PROCESS..



bartosz.galek 2 months ago

[setup-gradle migration]

We are pass 50% of migration done! (883 PR's merged!)

Please do not force me to "force merge" those PR in your repositories! It will take 5 minutes of your time!

Some of you already seen me direct contacting people, but this way is really painful 😉

So agai

bartosz.galek 1 month ago

[setup-gradle migration]

It's 676 PR's to go. We are ~60% done.

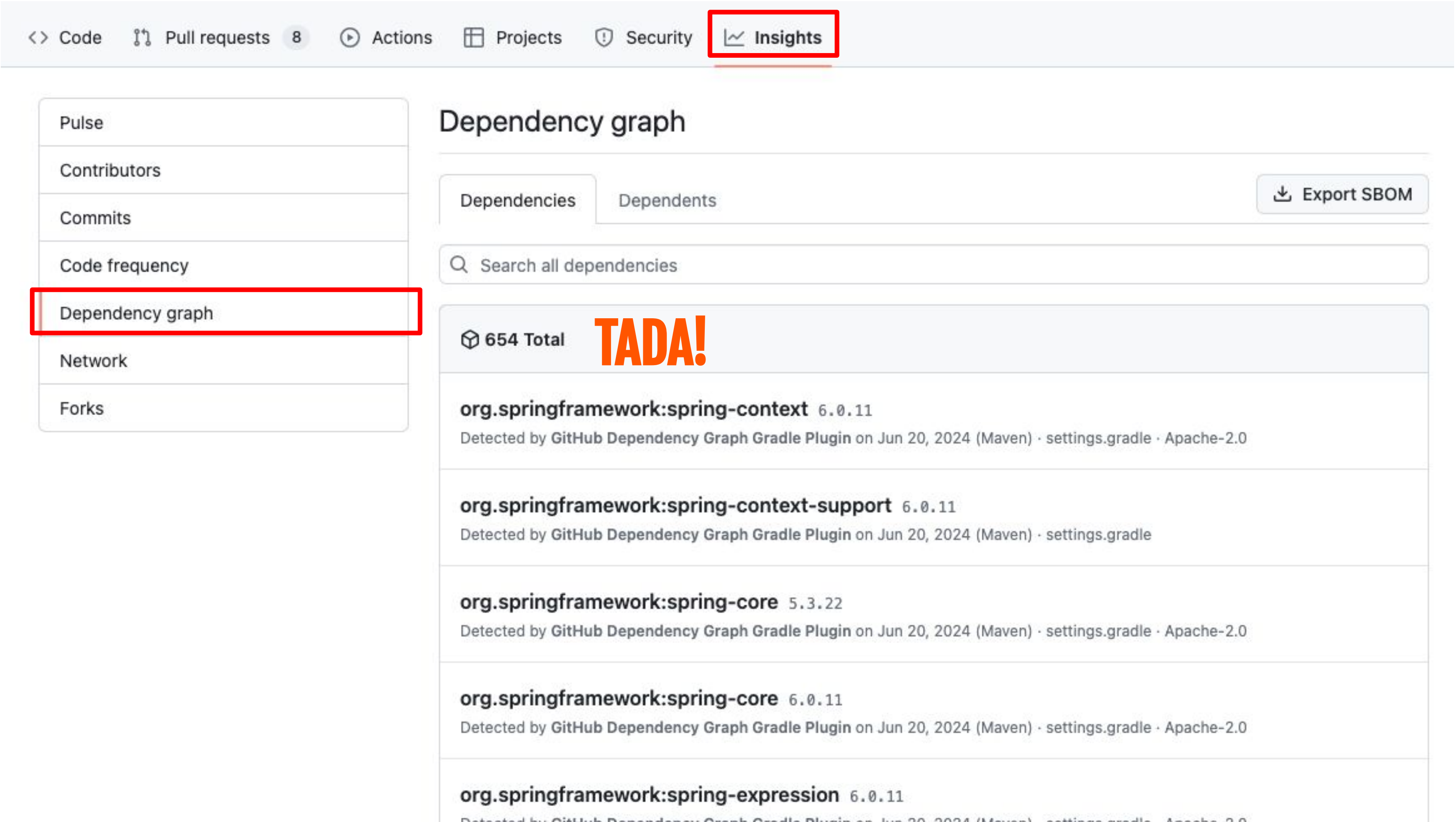
Please take care of `TPAS-5062 | Introduce setup-gradle action` pull requests!

Here is the full report of missing 676 pull requests, with responsible teams included https://docs.google.com/spreadsheets/d/1jdE-CLTaUwb9jyFcXAw3u3AsO07vP9-

We're about ~30% through, but we need more!

It will take you 10 seconds to review those PR's, please do it! 🙂

BUT WE GOT THERE!



Dependency graph

654 Total

TADA!

org.springframework:spring-context `6.0.11`
Detected by **GitHub Dependency Graph Gradle Plugin** on Jun 20, 2024 (Maven) · settings.gradle · Apache-2.0

org.springframework:spring-context-support `6.0.11`
Detected by **GitHub Dependency Graph Gradle Plugin** on Jun 20, 2024 (Maven) · settings.gradle

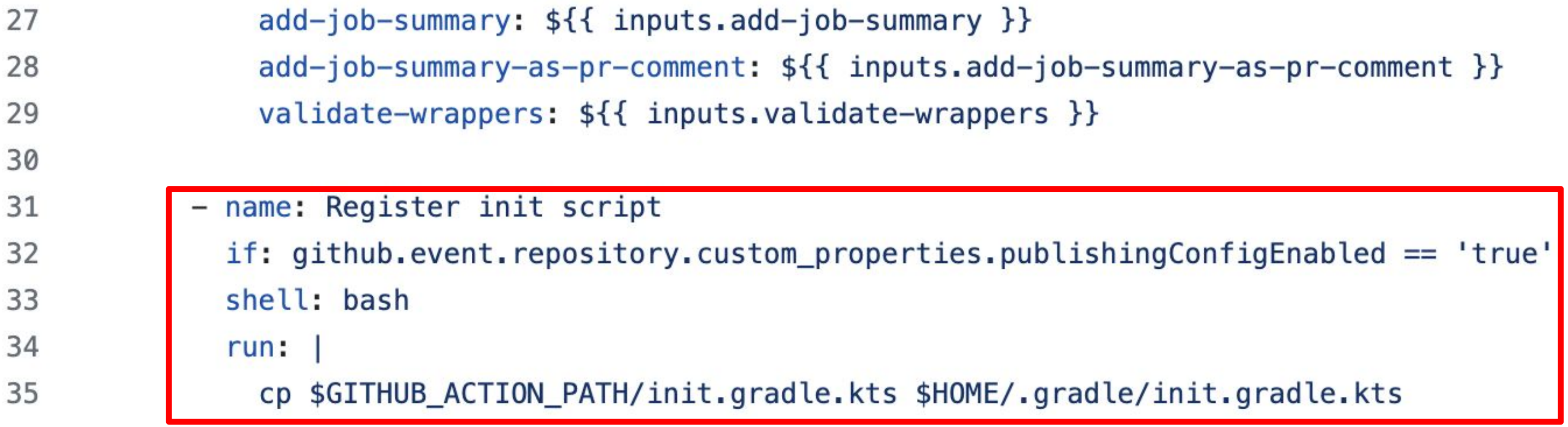
org.springframework:spring-core `5.3.22`
Detected by **GitHub Dependency Graph Gradle Plugin** on Jun 20, 2024 (Maven) · settings.gradle · Apache-2.0

org.springframework:spring-core `6.0.11`
Detected by **GitHub Dependency Graph Gradle Plugin** on Jun 20, 2024 (Maven) · settings.gradle · Apache-2.0

org.springframework:spring-expression `6.0.11`
Detected by **GitHub Dependency Graph Gradle Plugin** on Jun 20, 2024 (Maven) · settings.gradle · Apache-2.0

DID WE ACHIEVE GLOBAL GRADLE SETTINGS?

THAT WAS THE GRAND PLAN™



```
        add-job-summary: ${{ inputs.add-job-summary }}
        add-job-summary-as-pr-comment: ${{ inputs.add-job-summary-as-pr-comment }}
        validate-wrappers: ${{ inputs.validate-wrappers }}

    - name: Register init script
      if: github.event.repository.custom_properties.publishingConfigEnabled == 'true'
      shell: bash
      run: |
        cp $GITHUB_ACTION_PATH/init.gradle.kts $HOME/.gradle/init.gradle.kts
```

WHY ON EARTH WOULD YOU NEED A GLOBAL INIT GRADLE SCRIPT???

01

REPOSITORIES CONFIGURATION

Configure our internal maven repository

02

SETUP MAVEN PUBLICATION

Remove duplication

03

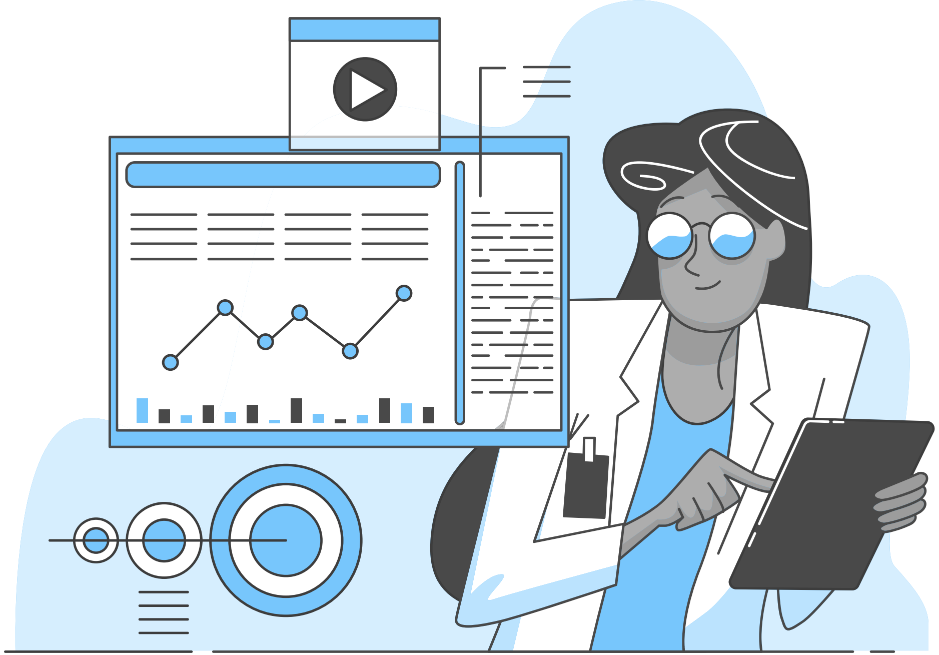
ENFORCE PLUGINS

We can enable internal plugins for all builds easily

04

ADDITIONAL BUILD LOGS

Improving developer experience

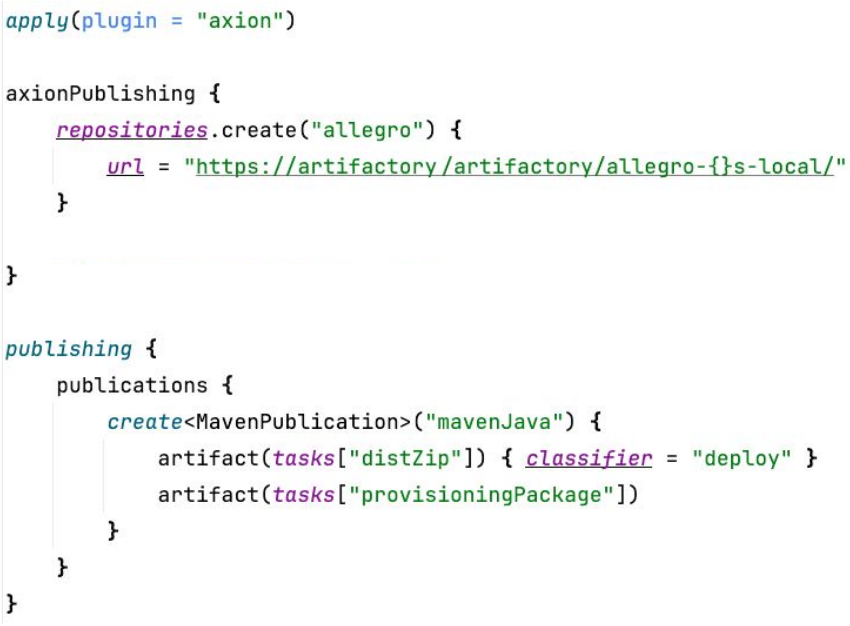


COMPARISON: THE OLD WORLD

INTERNAL GRADLE PLUGIN - AXION

Problems:

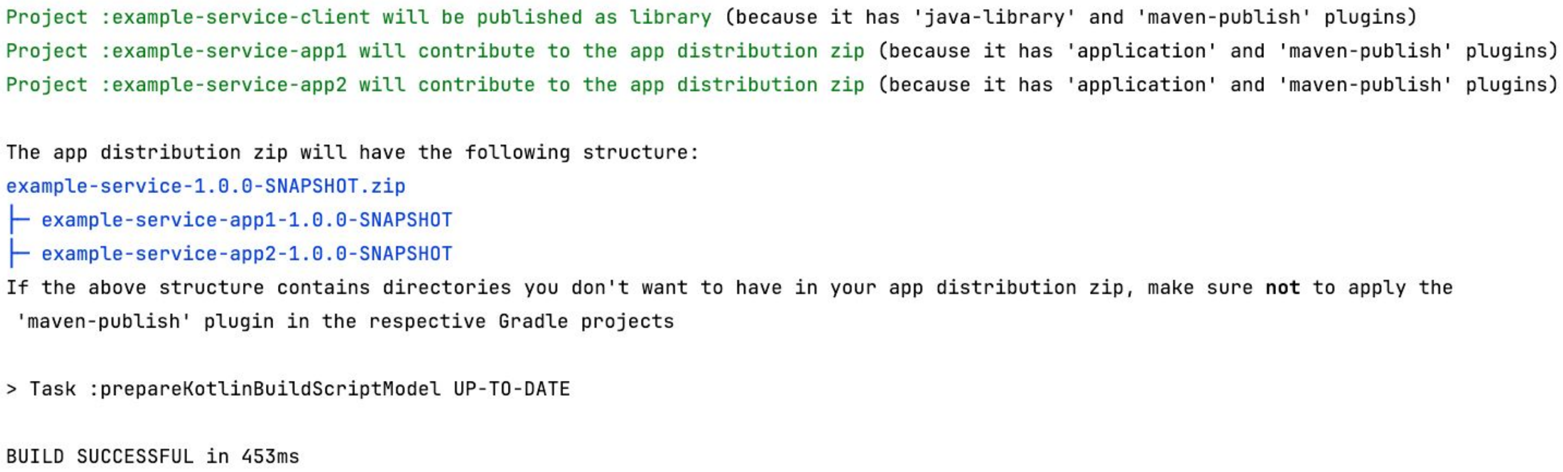
- copy-pasteable configuration which no one really understands
- changes require adoption throughout the entire organization (version bump is needed)
- nothing like any tutorial ;)



```
apply(plugin = "axion")

axionPublishing {
    repositories.create("allegro") {
        url = "https://artifactory/artifactory/allegro-{}s-local/"
    }

}

publishing {
    publications {
        create<MavenPublication>("mavenJava") {
            artifact(tasks["distZip"]) { classifier = "deploy" }
            artifact(tasks["provisioningPackage"])
        }
    }
}
```

COMPARISON: THE NEW WORLD

NO INTERNAL GRADLE PUBLICATION PLUGIN NEEDED!



```
Project :example-service-client will be published as library (because it has 'java-library' and 'maven-publish' plugins)
Project :example-service-app1 will contribute to the app distribution zip (because it has 'application' and 'maven-publish' plugins)
Project :example-service-app2 will contribute to the app distribution zip (because it has 'application' and 'maven-publish' plugins)

The app distribution zip will have the following structure:
example-service-1.0.0-SNAPSHOT.zip
├─ example-service-app1-1.0.0-SNAPSHOT
├─ example-service-app2-1.0.0-SNAPSHOT
If the above structure contains directories you don't want to have in your app distribution zip, make sure not to apply the
 'maven-publish' plugin in the respective Gradle projects

> Task :prepareKotlinBuildScriptModel UP-TO-DATE

BUILD SUCCESSFUL in 453ms
```

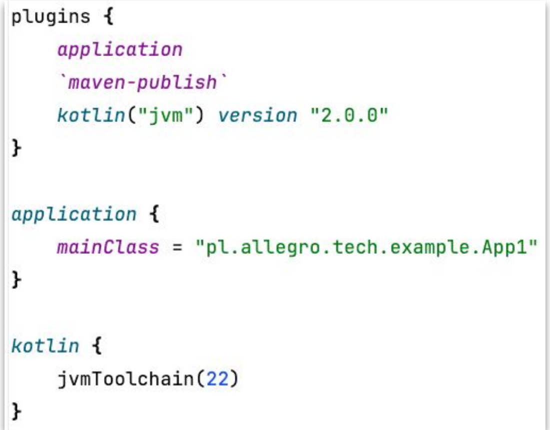
LIVE DEMO

I KNEW YOU WAITED
FOR IT :)

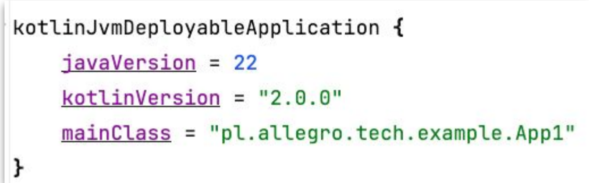
JUST WAIT FOR DECLARATIVE GRADLE!

WE ARE SO EXCITED!

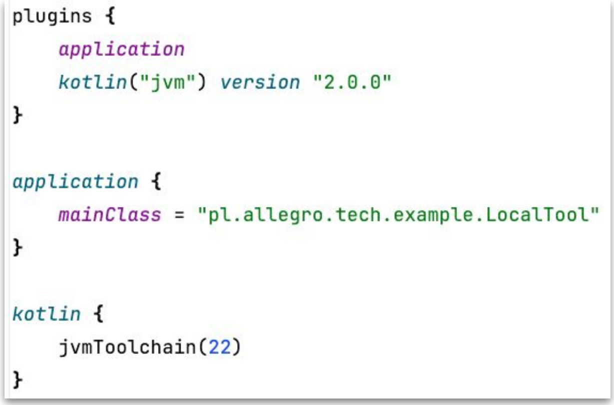
We hope that GitHub and other tools will then have an easier life with analysing build dependencies!



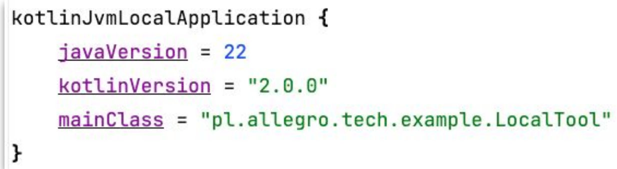
```
plugins {
    application
    `maven-publish`
    kotlin("jvm") version "2.0.0"
}

application {
    mainClass = "pl.allegro.tech.example.App1"
}

kotlin {
    jvmToolchain(22)
}
```



```
kotlinJvmDeployableApplication {
    javaVersion = 22
    kotlinVersion = "2.0.0"
    mainClass = "pl.allegro.tech.example.App1"
}
```



```
plugins {
    application
    kotlin("jvm") version "2.0.0"
}

application {
    mainClass = "pl.allegro.tech.example.LocalTool"
}

kotlin {
    jvmToolchain(22)
}
```



```
kotlinJvmLocalApplication {
    javaVersion = 22
    kotlinVersion = "2.0.0"
    mainClass = "pl.allegro.tech.example.LocalTool"
}
```

RESULTS

- migrated most of the codebase
- organization Dependencies reports have better coverage!
- almost nobody experienced any issues with the migration
  (we found a small issue in few repositories, but fixed it upstream!
  https://github.com/gradle/actions/pull/191/files)

PLANS

- enable gradle wrapper validation
- test out gradle-home-cache-cleanup option
- setup gradle-build-cache server company wide (configuration, host, default options)
- configure signing plugin in init script
- replace manual reporting with tools like FOSSA

CHALLENGES

- GitHub’s depenedabot security advisories sometimes cannot create automatic PR with fix, cause to transitive dependency found
- FOSSA tool does cool reporting and license verification BUT does not support loading SBOMS form GitHub (yet!)
- How to store maven credentials securely in GitHub runners

BEHIND THE SCENES

SPECIAL MENTIONS

This whole project wouldn't be possible without help of

- Wojciech Szewczuk (Allegro deployment platform team)
- Daz DeBoer (Gradle GitHub actions maintainer)!

Andamio Taskforce (Bartosz, Radek, Aleksandr, Michał)
will continue to fight for simplest build process possible ;)

Thanks!

Do you have any questions?